Computer Science

DEVELOPMENT ENVIRONMENT FOR TEACHING A FUNCTIONAL PROGRAMMING LANGUAGE: FUNCTIONAL NETWORKED INTEGRATED ENVIRONMENT (FUNNIE)

Brian Howard(*), Computer Science Department, DePauw University, IN 46135 (bhoward@depauw.edu)
Christopher Colvard, Computer Science Department, DePauw University, IN 46135 (ccolvard@depauw.edu)
Sara Pagliaro, Computer Science Department, Siena College, NY 12211 (sara.pagliaro@students.siena.edu)

The long-term aim of the Functional Networked Integrated Environment (FUNNIE) project is to develop a programming environment specifically tuned to the needs of students and instructors, based on a modern high-level functional language such as Haskell. The emphasis is placed on producing tools that are easy to use for students doing coursework for a course, rather than being tuned for professional software development. Existing systems like BlueJ, DrScheme, Vital, and Helium try to accomplish this, but are still not very attuned to the needs of beginner students and lack features geared toward class usage and participation. In order for the environment to meet its goal of aiding in the learning process, it should include tools for editing, running, and debugging programs, visualizing data and program structures, managing projects, accessing documentation, as well as allowing networked project collaboration.

The software for FUNNIE is developed in Java, with the user interface incorporating components of the Java Swing classes. It should work on all Java-compatible operating systems running at least the Java 2 Platform, Standard Edition, version 1.3. The networking feature is accomplished with user-level TCP sockets, requiring no dedicated server hardware.

In order to adapt FUNNIE to the needs of beginner students, its language needs to be easy to learn and utilize. After evaluating several existing systems and their languages, we created our own functional programming language named HasCl, essentially a subset of Haskell with some syntax influenced by Clean and C. HasCl supports calculations on a broad range of numbers, characters, string, and booleans, as well as tuple and list types for compound values. The basic unit of a program is the function, which may be defined using recursion and pattern matching. Future extensions of HasCl will include user-defined types.

Most of FUNNIE's user interface is standard for functional programming language environments. It allows a client to load in existing definitions and to create new ones, to track the history of created definitions, and to execute expressions. However unlike most typical functional programming language environments, FUNNIE also aids its users with easier debugging techniques. It allows users to step through expressions similarly to the stepper in DrScheme.

Learning can also be enhanced through group partnerships because it allows for peer-to-peer instruction. FUNNIE supports group activities and assignments; it is a networked environment so that a whole class can work on one program together and assist each other in learning more about the language and environment. The current version of FUNNIE supports basic functionality in this area, allowing users to log-in and share definitions via a class moderator.

The initial version of FUNNIE implements the basic ideas of cooperative learning for novice students. In the future, FUNNIE should expand on the basic operations it currently performs; for instance, it should allow graphics to be displayed or generated MIDI music to be played by executing an expression. New definition libraries can be written for HasCl. FUNNIE still has some vague error messages like the other existing systems, so in the future it should display more comprehensible error messages to help the user understand why his or her error occurred. FUNNIE should also be able to facilitate small group interaction within a class setting, expanding on the same ideas as the current class collaboration.

Acknowledgement: This work was supported by National Science Foundation Grant Number EIA-0242293.